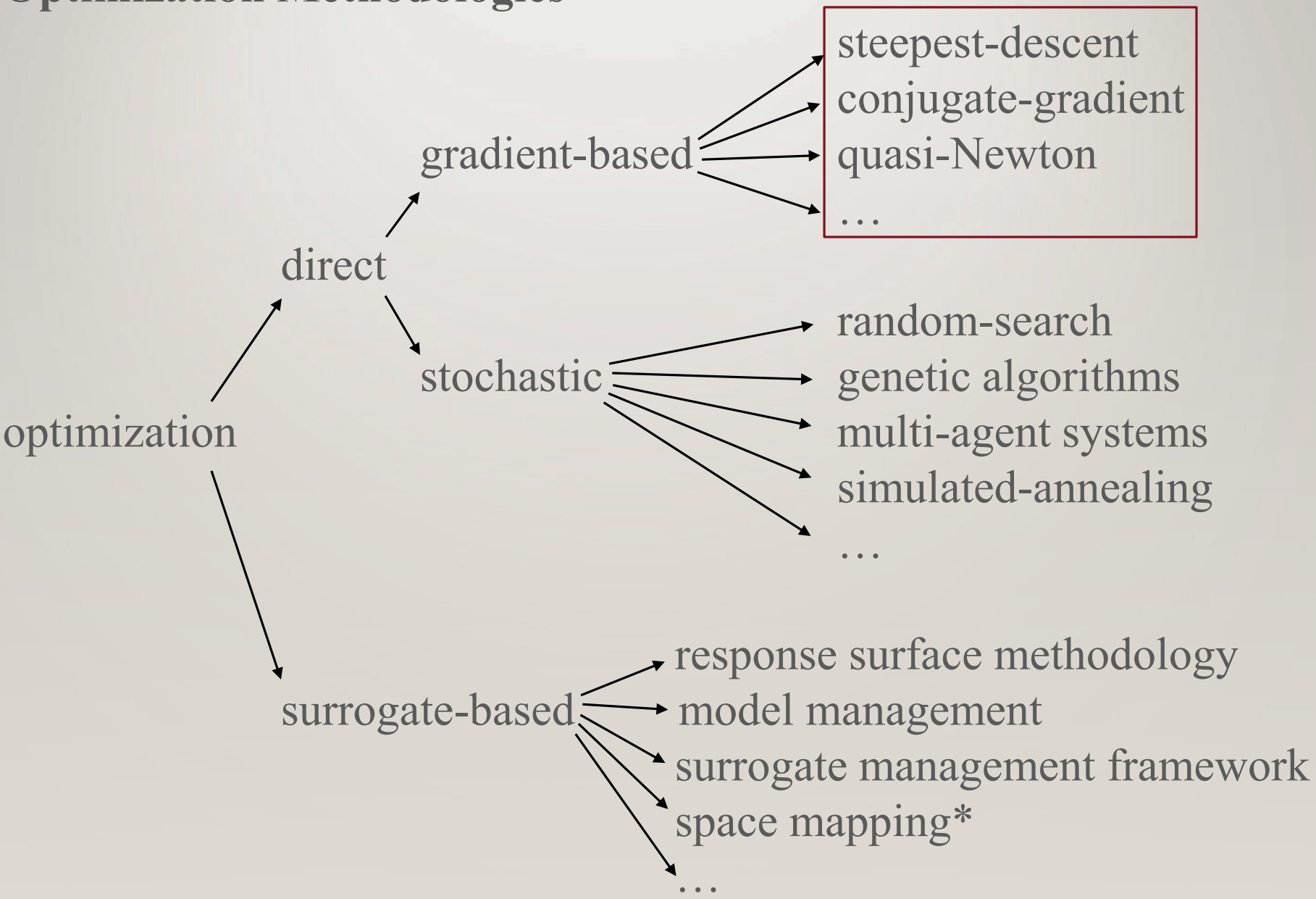
NONLINEAR OPTIMIZATION

Qingsha Cheng 程庆沙

HOMEWORK

- **Cover page with course name, your name and ID**
- **Printed Matlab Code with your name and key comments**
- **Screenshot of Results**
- **Zip file of your Matlab code**

Optimization Methodologies



Gradient-Based Unconstrained Optimization

Conditions for a local minimizer

Descent methods

Line search

Steepest descent methods

Conjugate gradient methods

Newton-type methods

Quasi-Newton methods

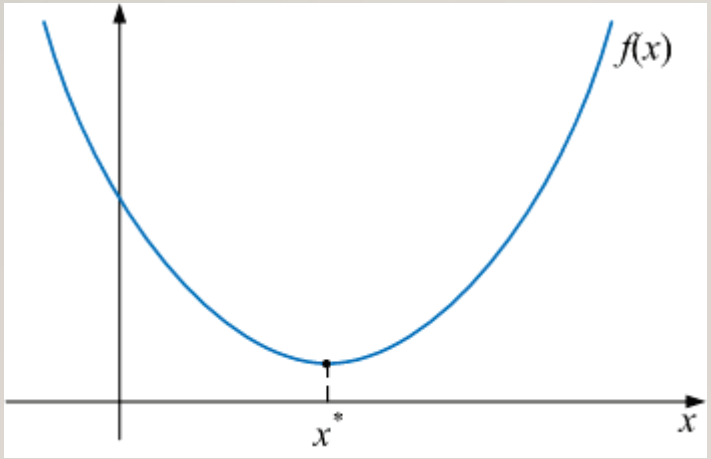
Optimization Problem

Consider the following unconstrained optimization problem

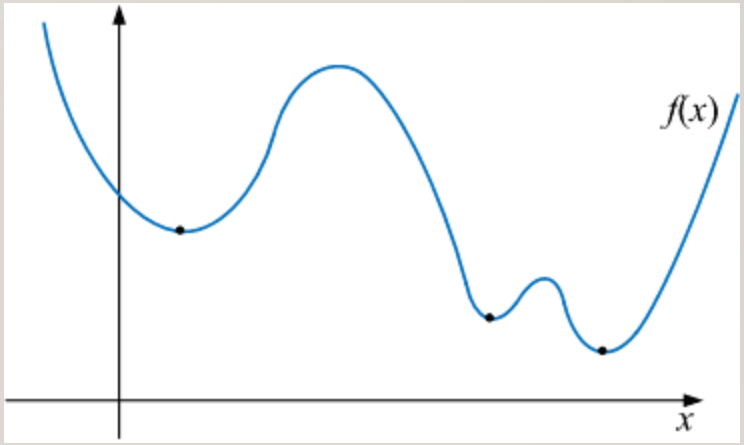
$$x^* = \arg\min_x f(x)$$

where $f: R^n \rightarrow R$; f is ***objective function***; x^* is the ***minimizer***.

The problem of maximizing a function can be reduced to a minimization problem of the function with opposite sign



Function with one global minimizer



Function with multiple local minimizers

Optimization Problem

During this lecture we shall consider methods that are able to find **local minimizer** of the objective function

Upon finding a local minimizer, one does not know whether it is a **global minimizer** or **one of many local ones**

There is **no** guarantee that the optimization method will find the local minimizer that is closest to the starting point

If **other local minimizers** must be explored, one can perform several runs with **different starting points**

Remark: During this lecture we will assume that function f has continuous derivatives up to a necessary order.

Conditions for a Local Minimizer

Definition: x^* is a local minimizer for $f: R^n \rightarrow R$ if

$$f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}) \quad \text{for} \quad \| \boldsymbol{x}^* - \boldsymbol{x} \| \leq \varepsilon \quad (\varepsilon > 0)$$

Taylor expansion of f at $\boldsymbol{x}$ gives:

$$f(\boldsymbol{x}+\boldsymbol{h}) = f(\boldsymbol{x}) + \boldsymbol{h}^T f'(\boldsymbol{x}) + O(\| \boldsymbol{h} \|^2)$$

where the gradient $f'(\boldsymbol{x})$ is defined as

$$f'(\boldsymbol{x}) = \left[\frac{\partial f}{\partial x_1}(\boldsymbol{x}) \quad \ldots \quad \frac{\partial f}{\partial x_n}(\boldsymbol{x}) \right]^T$$

Thus, if $\boldsymbol{x}$ is a local minimizer, there is no $\boldsymbol{h}$ such that $f(\boldsymbol{x}+\boldsymbol{h}) < f(\boldsymbol{x})$ for $\boldsymbol{h}$ small enough

Conditions for a Local Minimizer (FONC)

Theorem: If $\boldsymbol{x}^*$ is a local minimizer for $f: R^n \rightarrow R$, then $f'(\boldsymbol{x}^*) = 0$.

Definition: If $f'(\boldsymbol{x}_s) = 0$, then $\boldsymbol{x}_s$ is said to be a *stationary point* for f.

Stationary points may be local minimizer, local maximizer or none of them; in order to distinguish between them one needs second-order Taylor expansion:

$$f(\boldsymbol{x}+\boldsymbol{h}) = f(\boldsymbol{x}) + \boldsymbol{h}^T f'(\boldsymbol{x}) + \frac{1}{2}\boldsymbol{h}^T f''(\boldsymbol{x})\boldsymbol{h} + O(\|\boldsymbol{h}\|^3)$$

where the Hessian $f''(\boldsymbol{x})$ is defined as

$$f''(\boldsymbol{x}) = \left[\frac{\partial^2 f}{\partial x_i \partial x_j}(\boldsymbol{x})\right]$$

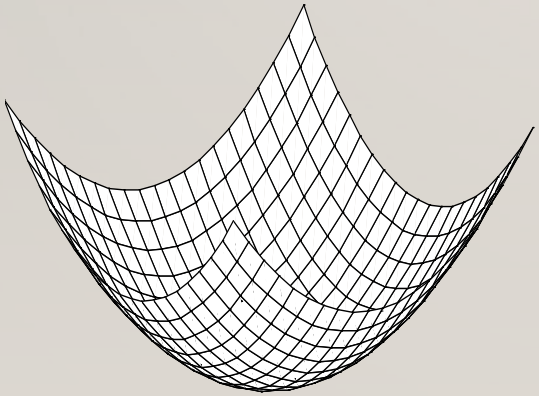
FONC is necessary for a local minimizer, but not sufficient (guaranteed).

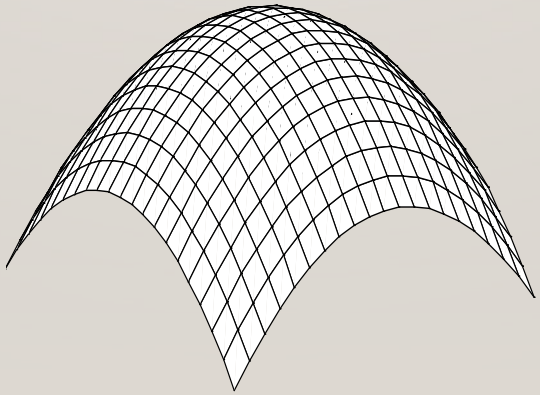
Conditions for a Local Minimizer (SONC)

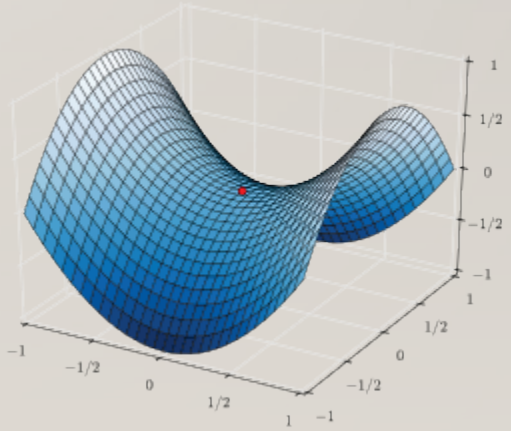
Theorem: If $\boldsymbol{x}$ is a local minimizer, then $f''(\boldsymbol{x})$ is positive semidefinite.

Remark: One of the tools used to determine whether an $n \times n$ matrix if positive definite is LU decomposition. We have: if $\boldsymbol{A}$ is symmetric and $\boldsymbol{A} = \boldsymbol{LU}$ is decomposition of $\boldsymbol{A}$, then $\boldsymbol{A}$ is positive definite if and only if $u_{ii} > 0$, $i = 1,...,n$.

Neighborhood of local minimizer, local maximizer and a saddle point







Necessary to satisfy both FONC and SONC. But not sufficient (guaranteed).

Click Picture to Play Video

Conditions for a Local Minimizer (SOSC)

Theorem: If x is a stationary point and $f''(\boldsymbol{x})$ is positive definite, then x is a local minimizer.

Theorem: If x is a stationary point and $f''(\boldsymbol{x})$ is positive semidefinite in the neighborhood of x, then x is a local minimizer

Theorem: If $\boldsymbol{x}$ is a stationary point and $f''(\boldsymbol{x}_s) \neq 0$. Then

1. If $f''(\boldsymbol{x})$ is positive definite, then x is a local minimizer
2. If $f''(\boldsymbol{x})$ is positive semidefinite, then x is a local minimizer or a saddle point
3. If $f''(\boldsymbol{x})$ is negative definite, then x is a local maximizer
4. If $f''(\boldsymbol{x})$ is negative semidefinite, then x is a local maximizer or a saddle point
5. If $f''(\boldsymbol{x})$ is neither definite nor semidefinite, then x is a saddle point

FONC and SOSC are sufficient (guarantee) for local minimizer, but not necessary. Minimizer does not necessarily satisfy SOSC. (Counter example $f(x,y)=x^2+y^2$)

Descent Methods

Just after learning the "steepest descent" method in optimization class...

Descent Methods

We will consider **iterative methods**, i.e., methods that produce sequence of vectors $\boldsymbol{x}^{(0)}$, $\boldsymbol{x}^{(1)}$, $\boldsymbol{x}^{(2)}$, ...

We want the sequence to **converge to a local minimizer** of a given objective function $f : R^n \rightarrow R$, i.e, $\boldsymbol{x}^{(k)} \rightarrow \boldsymbol{x}^*$ for $k \rightarrow \infty$

In most cases we try to enforce the **descending property**

$$f(\boldsymbol{x}^{(k+1)}) < f(\boldsymbol{x}^{(k)})$$

In ***descent methods***, a new vector, $\boldsymbol{x}^{(k+1)}$, is found starting from $\boldsymbol{x}^k$ and proceeding along so-called ***descent direction***, along which function f is decreasing (at least locally)

Descent Methods

Convergence rate of the iterative method describes how **quickly** we approach the **minimizer** x^*

Linear convergence:

$$||\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*|| \leq c_1 ||\boldsymbol{x}^{(k)} - \boldsymbol{x}^*|| \text{ with } 0 < c_1 < 1 \text{ and } \boldsymbol{x}^{(k)} \text{ close to } \boldsymbol{x}^*$$

Quadratic convergence:

$$||\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*|| \leq c_2 ||\boldsymbol{x}^{(k)} - \boldsymbol{x}^*||^2 \text{ with } 0 < c_2 < 1 \text{ and } \boldsymbol{x}^{(k)} \text{ close to } \boldsymbol{x}^*$$

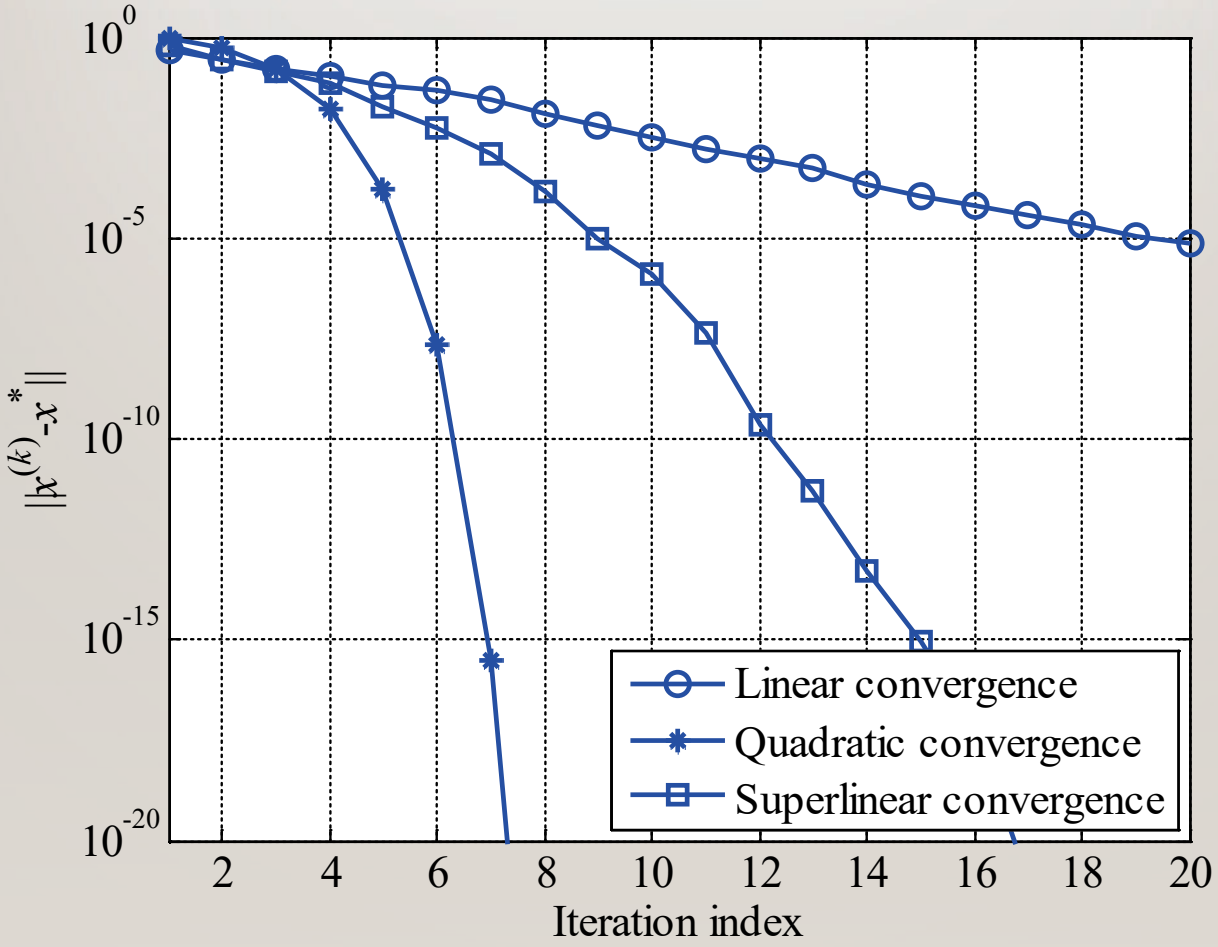
Superlinear convergence:

$$||\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*|| / ||\boldsymbol{x}^{(k)} - \boldsymbol{x}^*|| \to 0 \text{ for } k \to \infty$$

Superlinear convergence is better than a linear convergence but normally not as good as quadratic convergence

Descent Methods

Examples of linear, quadratic and superlinear convergence rate:



Descent Methods

Structure of a descent method

$k = 0$; $\boldsymbol{x} = \boldsymbol{x}^{(0)}$; *found* = false;
while ~*found* & $k <= k_{max}$
    $\boldsymbol{h}$ = search_direction($\boldsymbol{x}$);
    if isempty($\boldsymbol{h}$)
        found = true;
    else
        Find step length α;
        $\boldsymbol{x} = \boldsymbol{x} + \alpha\boldsymbol{h}$;
        $k = k + 1$;
        found = update(*found*);
    end
end

Descent Methods: Stopping Criteria

The following are stopping criteria used in practice

1.Convergence in argument

$$\| \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} \| < \varepsilon_1$$

2.Convergence in function value

$$\| f(\boldsymbol{x}^{(k+1)}) - f(\boldsymbol{x}^{(k)}) \| < \varepsilon_2$$

3.Vanishing of gradient

$$\| f'(\boldsymbol{x}^{(k)}) \| < \varepsilon_3$$

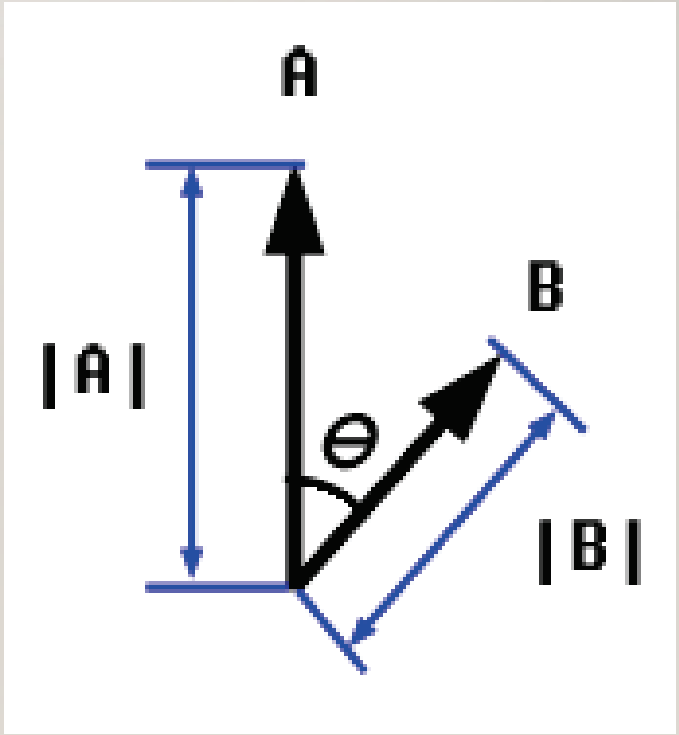
4.Convergence in function value by testing Hessian

$$\frac{1}{2}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)})^T f''(\boldsymbol{x}^{(k)})(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}) < \varepsilon_4$$

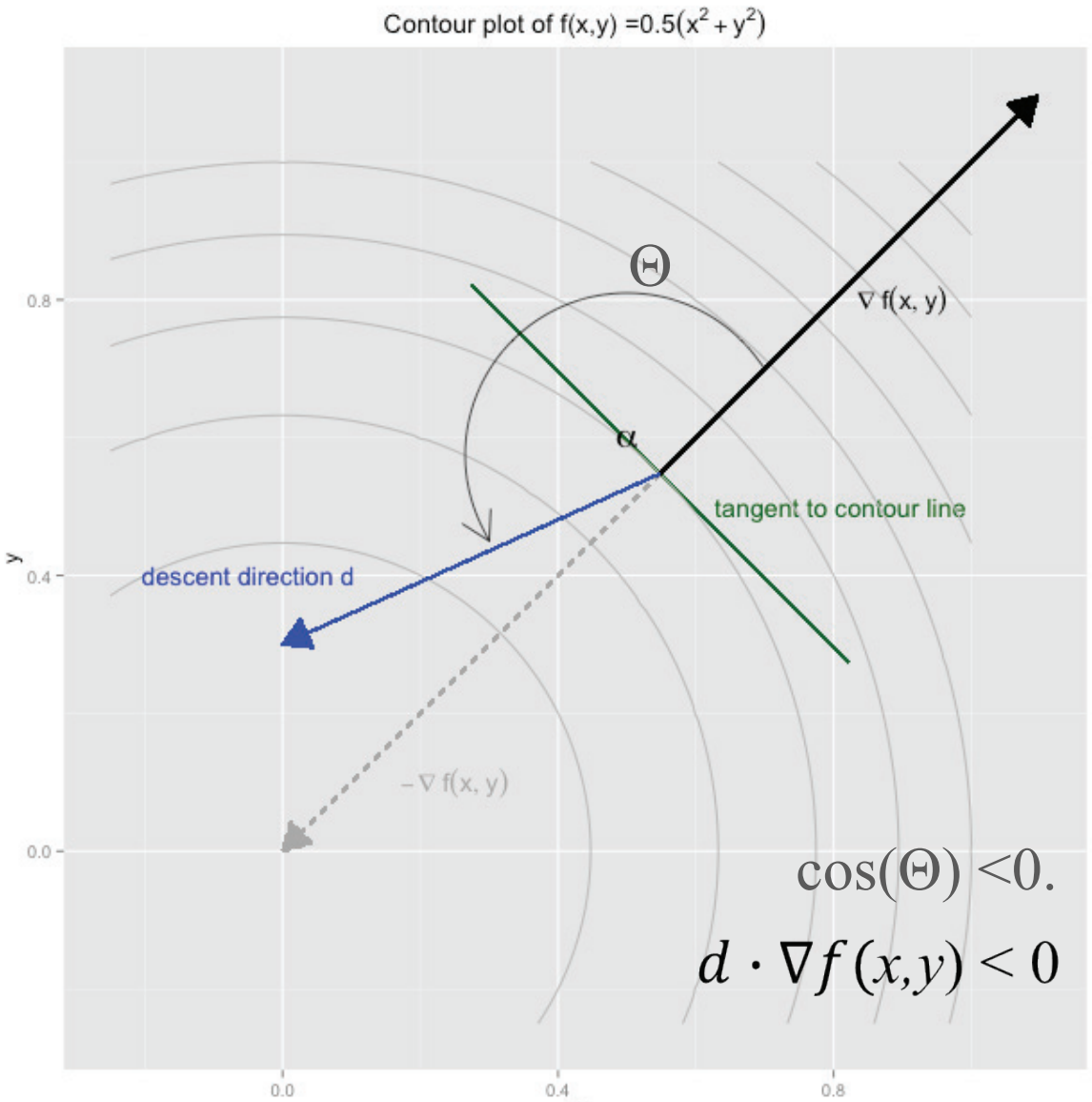
Descent Methods: Inner Product or Dot Product

$A \cdot B = A^T B = ||A||\ ||B||\ \cos(\Theta)$

1. If A and B are perpendicular (at $\Theta = 90^\circ$), $A^T B = 0$, because $\cos(\Theta) = 0$.
2. If $\Theta > 90^\circ$, $A^T B < 0$, because $\cos(\Theta) < 0$.
3. If $\Theta < 90^\circ$, $A^T B > 0$, because $\cos(\Theta) > 0$.



Descent Methods: Descent Direction



Descent Methods: Descent Directions

We have $f(\boldsymbol{x}+\alpha\boldsymbol{h}) = f(\boldsymbol{x}) + \alpha\boldsymbol{h}^T f'(\boldsymbol{x}) + O(\alpha^2)$, with $\alpha > 0$

For small α the first two terms are dominating
=> the sign of $\alpha\boldsymbol{h}^T f'(\boldsymbol{x})$ decides whether we go uphill or downhill

Definition: $\boldsymbol{h}$ is a descent direction from $\boldsymbol{x}$ if $\boldsymbol{h}^T f'(\boldsymbol{x}) < 0$

Let θ be the angle between $\boldsymbol{h}$ and $-f'(\boldsymbol{x})$ given by $\cos\theta = \dfrac{-\boldsymbol{h}^T f'(\boldsymbol{x})}{\|\boldsymbol{h}\| \cdot \|f'(\boldsymbol{x})\|}$

Definition: A method that uses search directions $\boldsymbol{h}^{(k)}$ such that corresponding $\theta < \pi/2 - \mu$ for all k, with $\mu > 0$ independent of k is called *absolute descent method*.

Descent Methods: Useful Result

The following result will be used later:

Theorem: If $f'(\boldsymbol{x}) \neq 0$ and $\boldsymbol{B}$ is a symmetric, positive definite matrix, then $\boldsymbol{h}^{(1)} = -\boldsymbol{B}f'(\boldsymbol{x})$ and $\boldsymbol{h}^{(2)} = -\boldsymbol{B}^{-1}f'(\boldsymbol{x})$ are descent directions.

Descent Methods with Line Search

Having chosen the search direction $\boldsymbol{h}$, **we need to decide how far we go**

Consider a function $\varphi(t) = f(\boldsymbol{x}+t\boldsymbol{h})$ with fixed $\boldsymbol{x}$ and $\boldsymbol{h}$

Taylor expansion of φ: $\varphi(t) = f(\boldsymbol{x}) + t\boldsymbol{h}^T f'(\boldsymbol{x}) + \frac{1}{2}t^2\boldsymbol{h}^T f''(\boldsymbol{x})\boldsymbol{h} + O(t^3)$

Also, $\varphi'(0) = \boldsymbol{h}^T f'(\boldsymbol{x}) < 0$ (assuming descent direction)

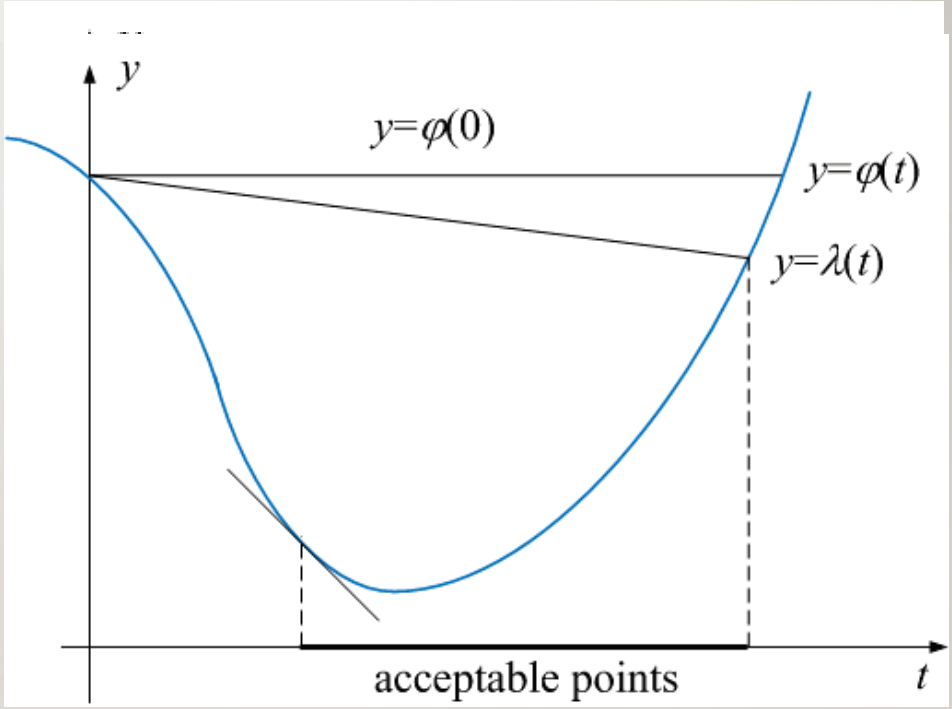
We stop the line search at t_s that should satisfy $\varphi'(t_s) < \varphi'(0)$

Practical conditions:

1. $\varphi(t_s) < \lambda(t_s)$, where $\lambda(t) = \varphi(0) + \rho \cdot \varphi'(0) \cdot t$, with $0 < \rho < 0.5$ (*)
(typically ρ is small, e.g., 0.001)
2. $\varphi'(t_s) \geq \beta \cdot \varphi'(0)$ with $\rho < \beta < 1$ (**)

Descent Methods with Line Search

Illustration of acceptable points:



Theorem: Consider an absolute descent method with the line search satisfying the two conditions shown in the previous slide. Then, if $f'(\boldsymbol{x})$ exists and is uniformly continuous for all $\boldsymbol{x}$ such that $f(\boldsymbol{x}) < f(\boldsymbol{x}^{(0)})$, then for $k \to \infty$:
either $f'(\boldsymbol{x}^{(k)}) = 0$ for some k, or $f(\boldsymbol{x}^{(k)}) \to -\infty$, or $f'(\boldsymbol{x}^{(k)}) \to 0$.

Descent Methods with Line Search

Soft versus exact line search:

Line search described before is called ***soft line search***

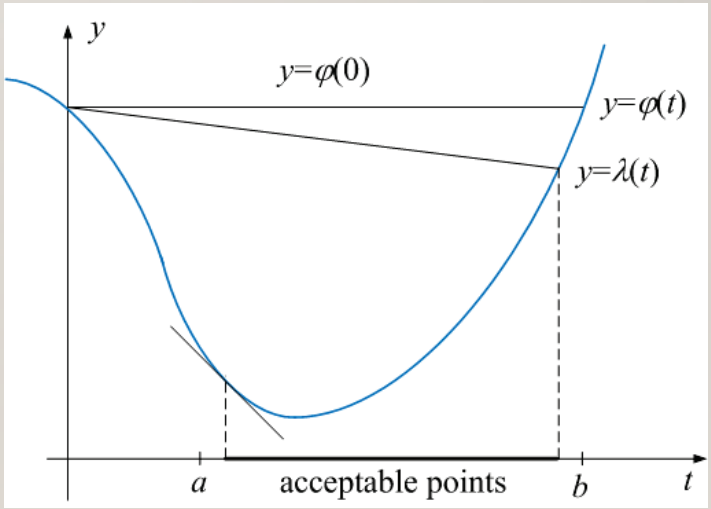
Exact line search seeks for $t_e = \arg\min_{t>0} f(\boldsymbol{x}+t\boldsymbol{h})$; necessary condition on t_e is either $f'(\boldsymbol{x}+t_e\boldsymbol{h}) = 0$ (stationary point for f) or else $\varphi'(t_e) = 0$ which means that $f'(\boldsymbol{x}+t_e\boldsymbol{h}) \perp \boldsymbol{h}$

Exact line search was dominant in early days of optimization, currently soft line search prevails because it is faster and even if the methods using it typically require more iterations to converge, the total number of function evaluations is smaller

Descent Methods with Line Search

Soft line search algorithm (first find $[a,b]$ containing acceptable points, then refine search within $[a,b]$):

if $\varphi'(0) \geq 0$
  $t = 0$;
else
  $k = 0$; $\gamma = \beta * \varphi'(0)$; $a = 0$; $b = \min\{1, t_{max}\}$;
  while $(\varphi(b) \leq \lambda(b))$ & $(\varphi'(b) \leq \gamma)$ & $(b < t_{max})$ & $(k < k_{max})$
    $k = k + 1$; $a = b$;
    $b = \min\{2b, t_{max}\}$
  end
  $t = b$;
  while $((\varphi(t) > \lambda(t))$ || $(\varphi'(t) < \gamma))$ & $(k < k_{max})$
    $k = k + 1$;
    Refine t and $[a,b]$;
  end
  if $\varphi(t) \geq \varphi(0)$
    $t = 0$;
  end
end



Interval $[a,b]$ containing acceptable points

Descent Methods with Line Search

Refinement algorithm (takes interval $[a,b]$ as an input; returns refined a and b as well as the new value of t):

$D = b - a$; $c = (\varphi(b) - \varphi(a) - D^*\varphi'(a))/D^2$; (*)
if $c > 0$
  $t = a - \varphi'(a)/(2c)$;
  $t = \min\{\max\{t, a + 0.1D\}, b - 0.1D\}\}$;
else
  $t = (a + b)/2$;
end
if $\varphi(t) < \lambda(t)$
  $a = t$;
else
  $b = t$;
end

(*) the quadratic function $q(t) = \varphi(a) + \varphi'(a)\cdot(t - a) + c\cdot(t - a)^2$ satisfies $q(a) = \varphi(a)$, $q'(a) = \varphi'(a)$ and $q(b) = \varphi(b)$; if $c > 0$ then q has a minimum and we set t to be the minimizer, otherwise we take t as the middlepoint of $[a,b]$

Descent Methods with Line Search

Exact line search: yields t_s which is sufficiently close to t_e (true minimizer of φ)

The algoritm for the exact line search can be the same as the for the soft search with the refinement loop changes from

while $((\varphi(t) > \lambda(t)) \,||\, (\varphi'(t) < \gamma)) \,\&\, (k < k_{max})$

to

while $(|\varphi'(t)| > \tau^*|\varphi'(0)|) \,\&\, (b - a > \varepsilon) \,\&\, (k < k_{max})$

with ε and τ being the small positive numbers indicating the tolerated error level

Line Search Methods

Textbook: Nonlinear Optimization in Electrical Engineering with Applications in MATLAB
P. 101-P.130, Chapter 4

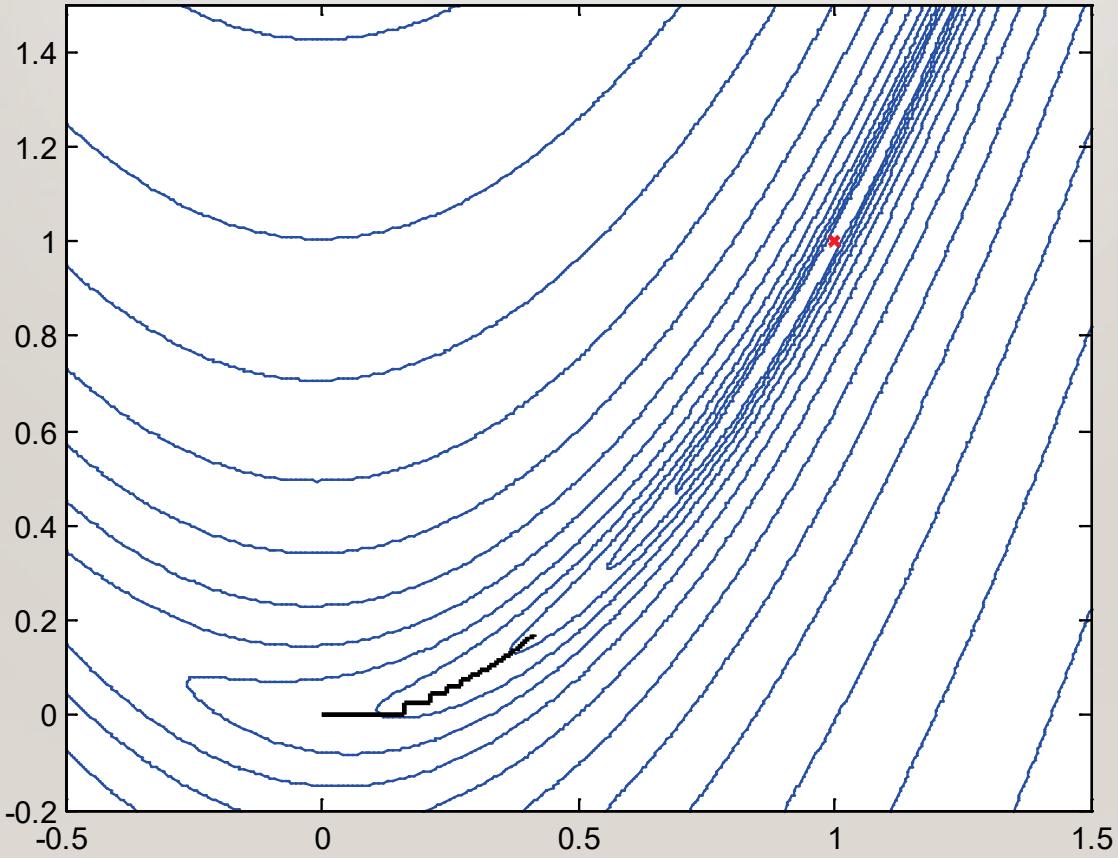
Steepest Descent Method

The steepest descent method uses, in each iteration, the search direction $h_{sd} = -f'(x)$

Although h_{sd} is the direction of a ***steepest descent***, practical performance of a steepest descent method can be very poor, especially if the exact line search is used => this method is not used in practice

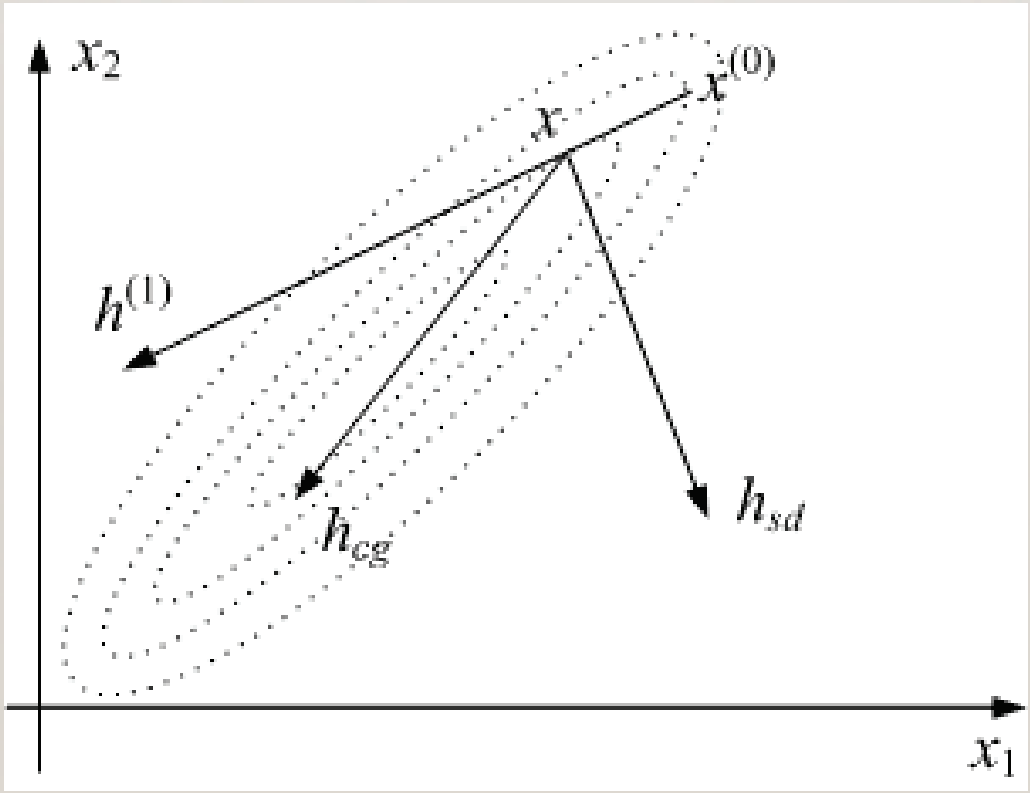
Steepest Descent Method: Example

Example: steepest descent method minimizing the Rosenbrock function: $f(x,y) = (1\text{-}x)^2 + 100(y - x^2)^2$



Conjugate Gradient Methods

Definition: A set of directions corresponding to vectors $\boldsymbol{h}^{(1)}$, $\boldsymbol{h}^{(2)}$, ..., is said to be ***conjugate*** with respect to a symmetric positive definite matrix $\boldsymbol{A}$ if $(\boldsymbol{h}^{(i)})^T \boldsymbol{A} \boldsymbol{h}^{(j)} = 0$ for all $i \neq j$.



Conjugate Gradient Methods

Example: consider a quadratic function $q(x) = a + b^T\boldsymbol{x} + 0.5\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x}$, where $\boldsymbol{H}$ is a positive definite matrix.

Let the first search step be in the descent direction $\boldsymbol{h}^{(1)}$ so that we are at $\boldsymbol{x}$ after **the exact line search**

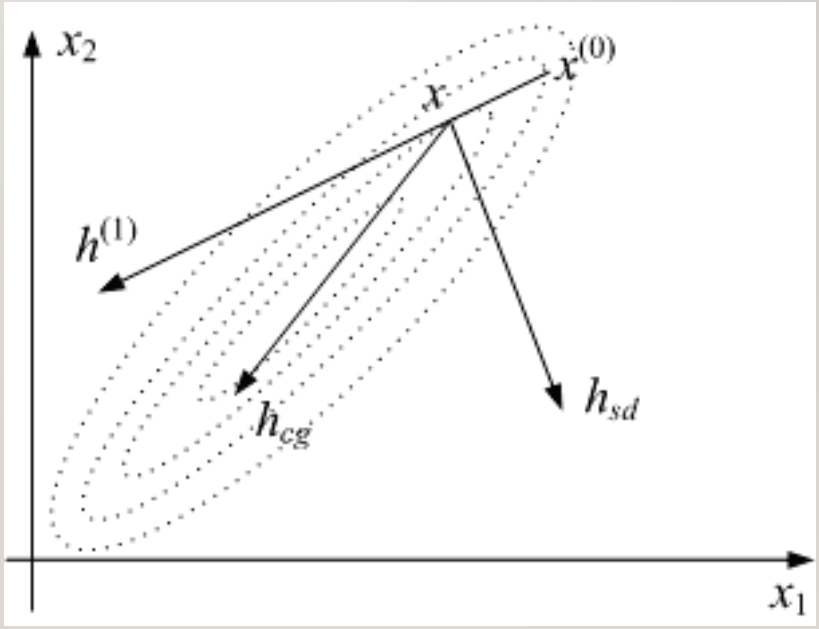
=> $\boldsymbol{h}^{(1)} \perp \boldsymbol{h}_{sd}$ (steepest descent direction)

=> $(\boldsymbol{h}^{(1)})^T(-q'(x)) = (\boldsymbol{h}^{(1)})^T(-\boldsymbol{b}-\boldsymbol{H}\boldsymbol{x}) = 0$

minimizer $\boldsymbol{x}^*$ satisfies $\boldsymbol{H}\boldsymbol{x}^*+\boldsymbol{b} = 0$

=> $(\boldsymbol{h}^{(1)})^T\boldsymbol{H}(\boldsymbol{x}^*-\boldsymbol{x}) = 0$

=> direction $\boldsymbol{x}^*-\boldsymbol{x}$ is conjugate to $\boldsymbol{h}^{(1)}$ with respect to $\boldsymbol{H}$



It can be shown that the conjugate direction always "lies" between $\boldsymbol{h}^{(1)}$ and $\boldsymbol{h}_{sd}$

Conjugate Gradient Methods

Quadratic models:

Function f is approximated locally using a quadratic function q of the form

$$q(\boldsymbol{x}) = a + \boldsymbol{b}^T \boldsymbol{x} + \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H} \boldsymbol{x}$$

We have $q'(x) = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{b}$ and $q''(x) = \boldsymbol{H}$

If $\boldsymbol{H}$ is positive definite, then q has a unique minimizer $\boldsymbol{x}^* = -\boldsymbol{H}^{-1}\boldsymbol{b}$

The ***quadratic termination*** holds for methods that find the exact minimum of the quadratic model in a finite number of steps

The steepest descent method does not have quadratic termination

Conjugate Gradient Methods

Conjugate gradient method algorithm:

$x = \boldsymbol{x}^{(0)}$; $k = 0$; *found* = false; $\gamma = 0$; $\boldsymbol{h}_{cg} = 0$
while ~*found* & $k \le k_{max}$
  $\boldsymbol{h}_{prev} = \boldsymbol{h}_{cg}$; $\boldsymbol{h}_{cg} = -f'(\boldsymbol{x}) + \gamma * \boldsymbol{h}_{prev}$;
  if $f'(\mathbf{x})^T \boldsymbol{h}_{cg} \ge 0$
    $\boldsymbol{h}_{cg} = -f'(\mathbf{x})$;
  end
  t = line_search($\boldsymbol{x}$,$\boldsymbol{h}_{cg}$); $\boldsymbol{x} = \boldsymbol{x} + t*\boldsymbol{h}_{cg}$;
  γ = ... (apply a suitable updating formula here)
  $k = k + 1$; *found* = ... (check the stopping criteria)
end

Remark: Recommended stopping criteria are:

1. $\|f'(\boldsymbol{x})\|_\infty \le \varepsilon_1$

or 2. $\|t\boldsymbol{h}_{cg}\|_2 \le \varepsilon_1(\varepsilon_2 + \|\boldsymbol{x}\|_2)$

Conjugate Gradient Methods

Theorem: Let the above algorithm be used with **exact line search** on **a quadratic model** with $\boldsymbol{x} \hat{I} \; R^n$ with iteration steps $\boldsymbol{h}^{(i)} = \boldsymbol{x}^{(i)} - \boldsymbol{x}^{(i-1)}$ corresponding to conjugate directions. Then

1. The search directions $\boldsymbol{h}_{cg}$ are downhill
2. The local gradient $f'(\boldsymbol{x}^{(k)})$ is orthogonal to $\boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)}, ..., \boldsymbol{h}^{(n)}$
3. The algorithm terminates after at most n steps

Remark: It is possible to prove global convergence of a conjugate gradient method with soft line search (note, that conjugate gradient method is *not* an absolute descent method)

Conjugate Gradient Methods: The Fletcher-Reeves Method

The Fletcher-Reeves method uses the following formula for γ

$$\gamma = \frac{f'(\boldsymbol{x})^T f'(\boldsymbol{x})}{f'(\boldsymbol{x}_{prev})^T f'(\boldsymbol{x}_{prev})}$$

Theorem: Suppose that the Fletcher-Reeves method with exact line search is applied to the quadratic function $q(\boldsymbol{x}) = a + b^T\boldsymbol{x} + 0.5\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x}$. If $f'(\boldsymbol{x}^{(k)}) \neq 0$ for $k = 1, 2, \ldots, n$, then the search directions $\boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)}, \ldots, \boldsymbol{h}^{(n)}$ are conjugate with respect to $\boldsymbol{H}$.

Remark: It is also possible to prove convergence of the Fletcher-Reeves method using soft line search.

Conjugate Gradient Methods: The Polak-Ribiere Method

The Polak-Ribiere method uses the following formula for γ

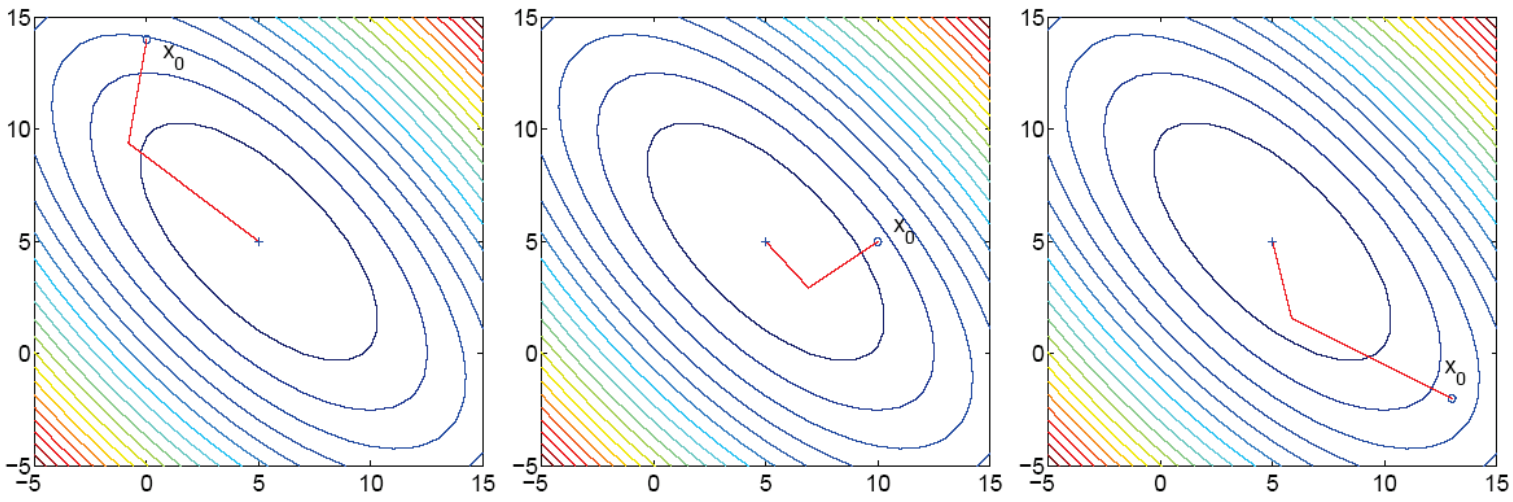
$$\gamma = \frac{(f'(\boldsymbol{x}) - f'(\boldsymbol{x}_{prev}))^T f'(B)}{f'(\boldsymbol{x}_{prev})^T f'(\boldsymbol{x}_{prev})}$$

Remark: Polak-Ribiere method is equivalent to Fletcher-Reeves method for a quadratic function because then we have $f'(\boldsymbol{x}_{prev}) \perp f'(\boldsymbol{x})$.

Remark: Polak-Ribiere method is globally convergent when used with exact line seach, however, convergence when using soft line search has not been proved so far.

CONJUGATE GRADIENT

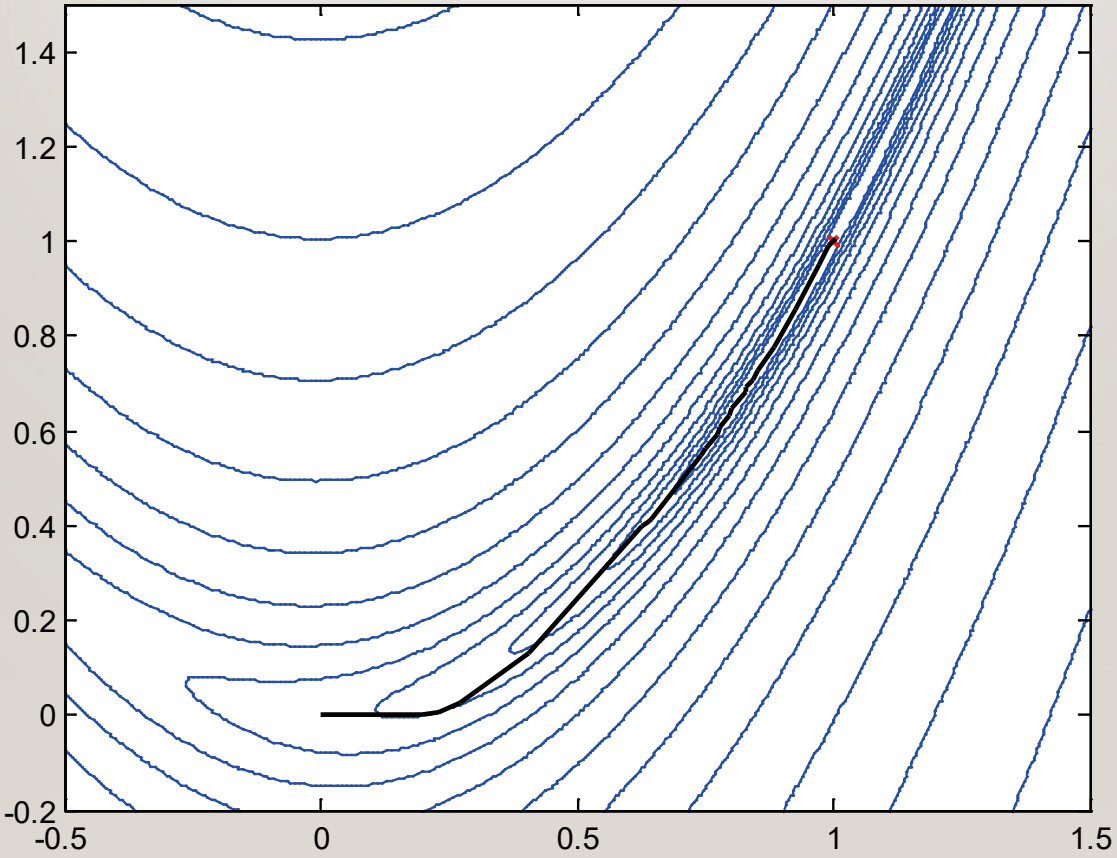
- An N-dimensional quadratic form can be minimized in at most N conjugate descent steps.



- 3 different starting points.
- Minimum is reached in exactly 2 steps.

Conjugate Gradient Methods: Example

Example: conjugate gradient method minimizing the Rosenbrock function: $f(x,y) = (1-x)^2 + 100(y - x^2)^2$



Conjugate Gradient Methods: Example

Soft versus exact line search: conjugate gradient method minimizing the Rosenbrock function: $f(x,y) = (1\text{-}x)^2 + 100(y - x^2)^2$

Termination condition: $\|f'(x)\|_\infty \leq 0.001$
line search parameters $r = 0.001$, $b = 0.8$

Line Search Type	Number of Iterations	Number of Function Evaluations
Exact	26	1336
Soft	61	472

Descent Methods with Trust Region

Consider Taylor expansion of f at x:

First order: $f(\boldsymbol{x}+\boldsymbol{h}) \cong q(\boldsymbol{h})$ with $q(\boldsymbol{h}) = f(\boldsymbol{x}) + \boldsymbol{h}^T f'(\boldsymbol{x})$

Second-order: $f(\boldsymbol{x}+\boldsymbol{h}) \cong q(\boldsymbol{h})$ with $q(\boldsymbol{h}) = f(\boldsymbol{x}) + \boldsymbol{h}^T f'(\boldsymbol{x}) + \frac{1}{2}\boldsymbol{h}^T f''(\boldsymbol{x})\boldsymbol{h}$

$q(\boldsymbol{h})$ is a good approximation to $f(\boldsymbol{x}+\boldsymbol{h})$ for sufficiently small $\|\boldsymbol{h}\|$

Consider the method that finds the new iteration step as

$$\boldsymbol{h}_{tr} = \arg\min_{\boldsymbol{h}\in D}\{q(\boldsymbol{h})\} \quad \text{where} \quad D = \{\boldsymbol{h} : \|\boldsymbol{h}\| \le \delta\},\ \delta > 0$$

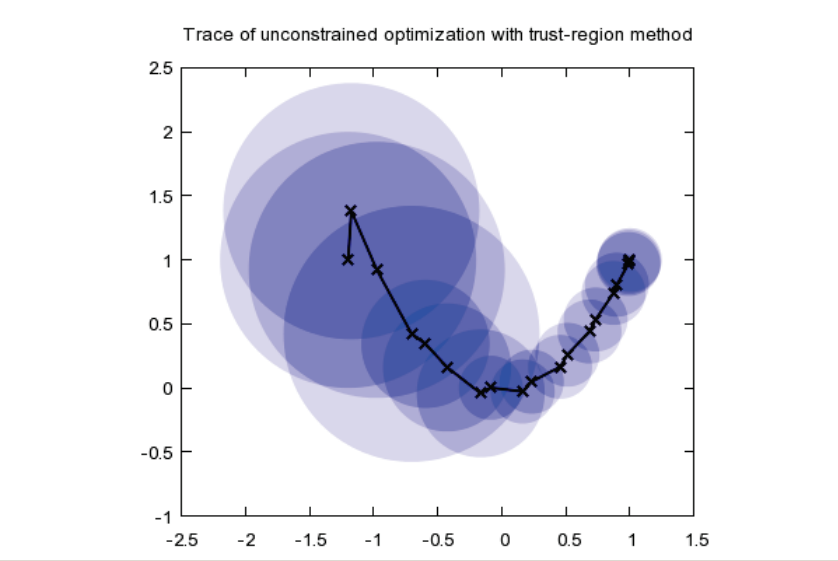
D is called the *trust region*; $\boldsymbol{h} = \boldsymbol{h}_{tr}$ is used as a candidate step and it is rejected if $f(x+\boldsymbol{h}) \ge f(\boldsymbol{x})$; the gain in objective function controls the size of the trust region for the next step; we use the following gain factor r:

$$r = \frac{f(\boldsymbol{x}) - f(\boldsymbol{x}+\boldsymbol{h})}{q(\boldsymbol{0}) - q(\boldsymbol{h})}$$

Descent Methods with Trust Region

Algorithm of a descent method with trust region:

$k = 0$; $x = x^{(0)}$; $\delta = \delta_0$; *found* = false;
while ~*found* & $k \le k_{max}$
  $k = k + 1$; h_{tr} = Minimizer of the model function q;
  $r = (f(x) - f(x + h_{tr}))/(q(0) - q(h_{tr}))$;
  if $r > 0.75$
    $\delta = 2^*\delta$;
  elseif $r < 0.25$
    $\delta = \delta/3$;
  end
  if $r > 0$
    $x = x + h_{tr}$;
  end
  Update *found*
end



Newton-Type Methods

Newton-Type Methods

Newton-type methods **do not use a line search** but find the next approximation of the solution based on the local properties of the objective function

Quadratic model of f in the neighborhood of $\boldsymbol{x}$:

$$f(\boldsymbol{x}+\boldsymbol{h}) \cong q(\boldsymbol{h}) \quad \text{with} \quad q(\boldsymbol{h}) = f(\boldsymbol{x}) + \boldsymbol{h}^T f'(\boldsymbol{x}) + \frac{1}{2}\boldsymbol{h}^T f''(\boldsymbol{x})\boldsymbol{h}$$

If $f''(\boldsymbol{x})$ is positive definite, then q has a unique minimizer at a point where the gradient of q is zero

Hence, we have: $f'(\boldsymbol{x}) + f''(\boldsymbol{x})\boldsymbol{h} = 0$

The search step can be obtained by solving the above equation

Newton's Method

Algorithm of the Newton's method:

$x = x^{(0)}$;
while stopping criteria not satisfied
    Solve $f''(x)h = -f'(x)$;
    $x = x + h$;
end

Theorem: If x is sufficiently close to a local minimizer x^* and $f''(x^*)$ is positive definite, then Newton's method is well defined and it converges quadratically towards x^*.

Newton's Method

Advantages of Newton's method:
1.Quadratically convergent from a good starting point if $f''(x)$ is positive definite
2.Simple and easy to implement

Disadvantages of Newton's method:
1.Not globally convergent for many problems
2.May converge towards a maximum or saddle point
3.The system of linear equations to be solved in each iteration may be ill-conditioned or singular
4.Requires analytic second order derivatives of f

Damped Newton's Methods

One of the most serious disadvantages of the Newton's method is that it does not work if $f''(\boldsymbol{x})$ is not positive definite; so-called damped Newton methods are designed to overcome this problem (I stands for the identity matrix):

solve $(f''(x) + \mu I)h_{dn} = -f'(x)$, $\mu \geq 0$
$x = x + \alpha h_{dn}$; $\alpha \geq 0$
adjust μ;

Thus, instead of finding the stationary point of a quadratic function, we find the search step as a stationary point of

$$q_\mu(\boldsymbol{h}) = q(\boldsymbol{h}) + \frac{1}{2}\mu \boldsymbol{h}^T \boldsymbol{h} = f(\boldsymbol{x}) + \boldsymbol{h}^T f'(\boldsymbol{x}) + \frac{1}{2}\boldsymbol{h}^T \left(f''(\boldsymbol{x}) + \mu \boldsymbol{I} \right) \boldsymbol{h}$$

Damped Newton's Methods

Remark: It can be shown that $f''(\boldsymbol{x}) + \mu \boldsymbol{I}$ is positive definite for sufficiently large μ

Remark: Damped Newton's method penalizes large steps (because of the term $0.5\mu \boldsymbol{h}^T \boldsymbol{h}$)

Remark: For very large μ we have $\boldsymbol{h} \cong -\mu^{-1} f'(\boldsymbol{x})$, i.e., we get a short step in the steepest descent direction

Theorem: If the matrix $f''(\boldsymbol{x})+\mu \boldsymbol{I}$ is positive definite, then

$$\boldsymbol{h}_{dn} = \arg \min_{\|\boldsymbol{h}\| \le \|\boldsymbol{h}_{dn}\|} \{q(\boldsymbol{h})\}$$

where $\boldsymbol{h}_{dn}$ is defined by the damped Newton's method and

$$q(\boldsymbol{h}) = f(\boldsymbol{x}) + \boldsymbol{h}^T f'(\boldsymbol{x}) + \frac{1}{2} \boldsymbol{h}^T f''(\boldsymbol{x}) \boldsymbol{h}$$

Damped Newton's Methods: Levenberg-Marquardt Method

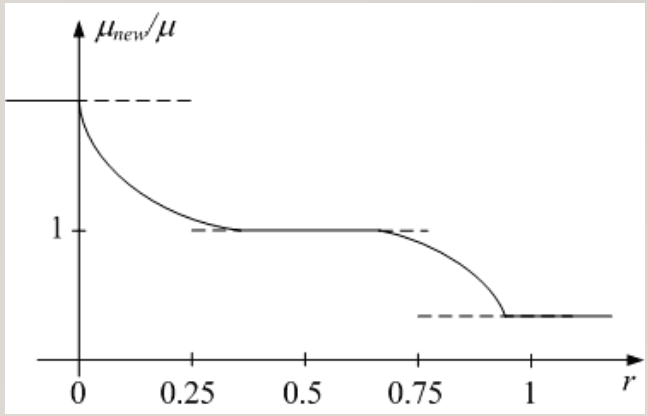
The most successful variant of damped Newton's method are so-called **Levenberg-Marquardt** type methods, where **parameter** μ is **updated** in **each iteration step** to ensure that $f''(\boldsymbol{x}) + \mu \boldsymbol{I}$ is positive definite; a trust region approach is also used to ensure sufficient decrease of the objective function

The gain factor used in Levenberg-Marquardt methods:

$$r = \frac{f(\boldsymbol{x}) - f(\boldsymbol{x}+\boldsymbol{h})}{q(\boldsymbol{0}) - q(\boldsymbol{h})}$$

Strategy for changing μ:

if $r > 0$
    $\mu = \mu * \max\{1/3, 1-2(2r-1)^3\}$
else
    $\mu = \mu * 2$



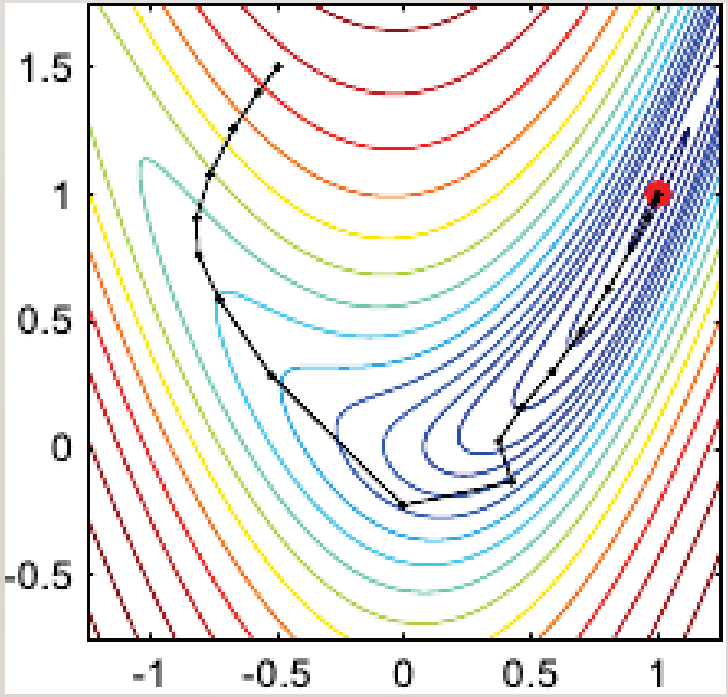
Damped Newton's Methods: Levenberg-Marquardt Method

Algorithm of the Levenberg-Marquardt method:

$x = x^{(0)}$; $\mu = \mu^{(0)}$; *found* = false; $k = 0$;
while ~*found* & $k \le k_{max}$
  while $f''(x)+\mu I$ not positive definite
    $\mu = 2*\mu$;
  end
  Solve $(f''(x)+\mu I)h_{dn} = -f'(x)$;
  Compute gain factor r;
  if $r > \delta$
    $x = x + h_{dn}$;
    $\mu = \mu*\max\{1/3, 1-2(2r-1)^3\}$;
  else
    $\mu = \mu*2$;
  end
  $k = k+1$; Update *found*;
end

Levenberg-Marquardt Method: Example

Example: Levenberg-Marquardt method minimizing the Rosenbrock function: $f(x,y) = (1-x)^2 + 100(y - x^2)^2$



Quasi-Newton Methods

Damped Newton methods alleviate the problems of the Newton methods, however, still require second derivatives of the cost function

Quasi-Newton methods: **approximation** of the second order derivative (or its inverse) is used instead of the Hessian matrix (or its inverse):

$$\boldsymbol{B} \cong f''(\boldsymbol{x}) \qquad \text{and} \qquad \boldsymbol{D} = f''(\boldsymbol{x})^{-1}$$

Approximation of inverse of Hessian is preferred because $f''(\boldsymbol{x})^{-1}$ is normally used to compute the search direction

Approximation matrices can be found using appropriate updating formulas

Quasi-Newton method can be used with a line search or trust region

Quasi-Newton Methods

Iteration step in quasi-Newton method with updating and line search:

Solve $Bh_{qn} = -f'(x)$; (or calculate $h_{qn} = -Df'(x)$;)
Line search along h_{qn} giving $h_{qn} = th_{qn}$; $x_{new} = x + h_{qn}$;
Update B (or update D)

Updating formula must satisfy the so-called quasi-Newton condition: $\boldsymbol{B}_{new}\boldsymbol{h}_{qn} = \boldsymbol{y} = f'(\boldsymbol{x}_{new}) - f'(\boldsymbol{x})$ (or $\boldsymbol{D}_{new}(f'(\boldsymbol{x}_{new}) - f'(\boldsymbol{x})) = \boldsymbol{h}_{qn}$); other conditions are also needed to get well defined matrices

Typically, we use $\boldsymbol{B}_{new} = \boldsymbol{B} + \boldsymbol{a}\boldsymbol{b}^T$ (rank-one updating formula) or $\boldsymbol{B}_{new} = \boldsymbol{B} + \boldsymbol{a}\boldsymbol{b}^T + \boldsymbol{u}\boldsymbol{v}^T$ (rank-two updating formula), where $\boldsymbol{a},\boldsymbol{b},\boldsymbol{u},\boldsymbol{v} \in R^n$

Quasi-Newton Methods: Updating Formulas

Broyden's rank-one formula (simple but symmetry and positive definiteness are not preserved):

$$\boldsymbol{B}_{new} = \frac{1}{\boldsymbol{h}_{qn}^T \boldsymbol{h}_{qn}} \left(\boldsymbol{y} - \boldsymbol{B}\boldsymbol{h}_{qn} \right) \boldsymbol{h}_{qn}^T \qquad \boldsymbol{D}_{new} = \boldsymbol{D} + \frac{1}{\boldsymbol{y}^T \boldsymbol{y}} \left(\boldsymbol{h}_{qn} - \boldsymbol{D}\boldsymbol{y} \right) \boldsymbol{y}^T$$

Symmetric updating (does not preserve positive definiteness and is numerically unstable)

$$\boldsymbol{D}_{new} = \boldsymbol{D} + \frac{1}{\boldsymbol{u}^T \boldsymbol{y}} \boldsymbol{u}\boldsymbol{u}^T \quad \text{with} \quad \boldsymbol{u} = \boldsymbol{h} - \boldsymbol{D}\boldsymbol{y}$$

$$\boldsymbol{B}_{new} = \boldsymbol{B} + \frac{1}{\boldsymbol{h}^T \boldsymbol{v}} \boldsymbol{v}\boldsymbol{v}^T \quad \text{with} \quad \boldsymbol{v} = \boldsymbol{y} - \boldsymbol{B}\boldsymbol{h}$$

Quasi-Newton Methods: Updating Formulas

DFP (Davidon-Fletcher-Powell) formula:

$$\mathbf{D}_{new} = \mathbf{D} + \frac{1}{\mathbf{h}^T\mathbf{y}}\mathbf{h}\mathbf{h}^T - \frac{1}{\mathbf{y}^T\mathbf{v}}\mathbf{v}\mathbf{v}^T$$

where $\boldsymbol{h} = \boldsymbol{x}_{new} - \boldsymbol{x}$, $\boldsymbol{y} = f'(\boldsymbol{x}_{new}) - f'(\boldsymbol{x})$, $\boldsymbol{v} = \boldsymbol{D}\boldsymbol{y}$

Features:

1. Preserves positive definite *D*-matrices if $\boldsymbol{h}_{qn}{}^T\boldsymbol{y} > 0$ in all steps.
2. Gives superlinear final convergence.
3. Gives global convergence for strictly convex functions provided that the line search is exact.

DFP formula may fail when working with soft line search

Quasi-Newton Methods: Updating Formulas

BFGS (Broyden-Fletcher-Goldfarb-Shanno) formula:

$$\boldsymbol{B}_{new} = \boldsymbol{B} + \frac{1}{\boldsymbol{h}^T \boldsymbol{y}} \boldsymbol{y}\boldsymbol{y}^T - \frac{1}{\boldsymbol{h}^T \boldsymbol{u}} \boldsymbol{u}\boldsymbol{u}^T$$

where $\boldsymbol{h} = \boldsymbol{x}_{new} - \boldsymbol{x}$, $\boldsymbol{y} = f'(\boldsymbol{x}_{new}) - f'(\boldsymbol{x})$, $\boldsymbol{u} = \boldsymbol{B}\boldsymbol{h}$

BFGS formula for D: $\boldsymbol{D}_{new} = \boldsymbol{D} + a_1 \boldsymbol{h}\boldsymbol{h}^T - a_2(\boldsymbol{h}\boldsymbol{v}^T + \boldsymbol{v}\boldsymbol{h}^T)$, where $\boldsymbol{h} = \boldsymbol{x}_{new} - \boldsymbol{x}$, $\boldsymbol{y} = f'(\boldsymbol{x}_{new}) - f'(\boldsymbol{x})$, $\boldsymbol{v} = \boldsymbol{D}\boldsymbol{y}$, $a_2 = (\boldsymbol{h}^T\boldsymbol{y})^{-1}$, $a_1 = a_2(1 + a_2(\boldsymbol{y}^T\boldsymbol{v}))$

BFGS formula with a soft line search gives convergence on convex functions

BFGS formula is always used with a soft line search and should be initiated with the full quasi-Newton step in each iteration (i.e., $t = 1$); line search should be "loose", e.g., $r = 10^{-4}$ and $b = 0.9$

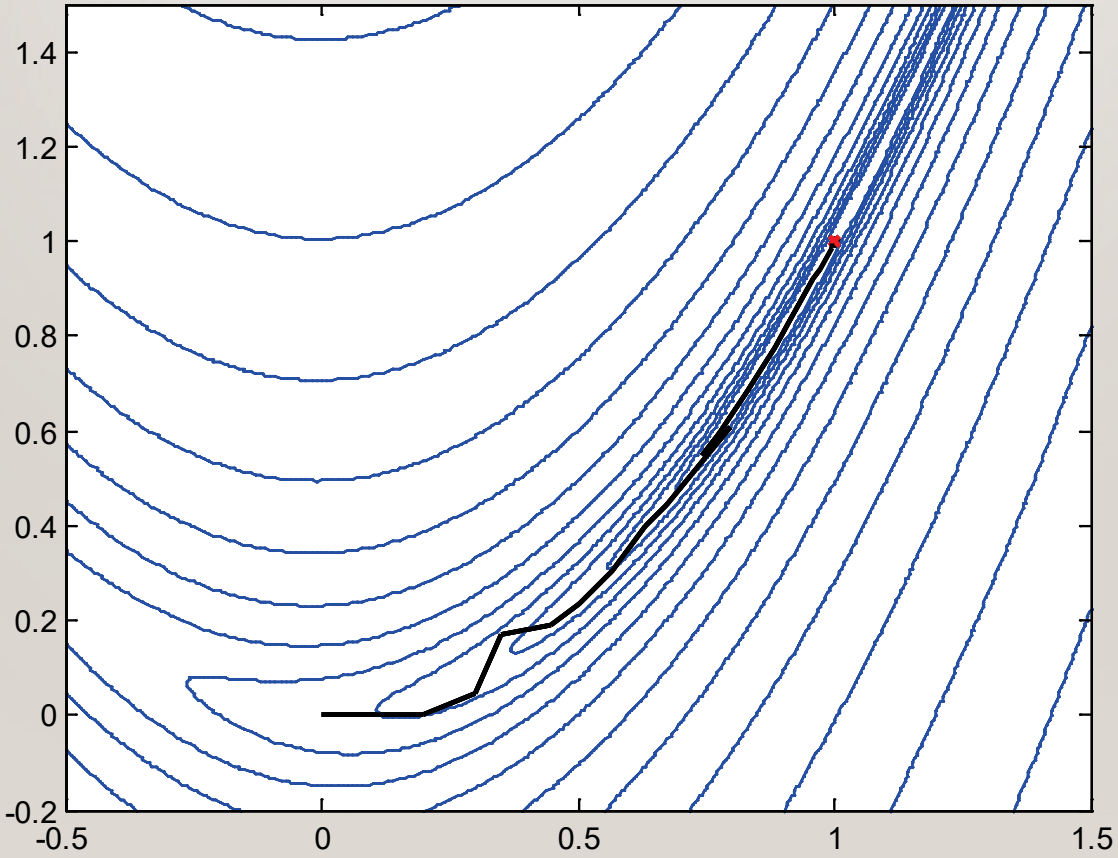
Quasi-Newton Methods: Algorithm

Algorithm of quasi-Newton method with BFGS updating:

$x = x^{(0)}$; $D = D^{(0)}$; $k = 0$; $n_f = 0$; % $D^{(0)} = I$ is recommended
while $\|f'(x)\| > \varepsilon$ & $k < k_{max}$ & $n_f < n_{f.max}$
  $h_{qn} = D^{-1}(-f'(x))$;
  $[t,k_f]$ = soft_line_search(x,h_{qn});
  $n_f = n_f + k_f$;
  $x_{new} = x + t \cdot h_{qn}$; $k = k + 1$;
  if $h_{qn}^T f'(x_{new}) > h_{qn}^T f'(x)$
    Update D;
  end
  $x = x_{new}$;
end

Quasi-Newton Method with BFGS Update: Example

Example: quasi-Newton method minimizing the Rosenbrock function: $f(x,y) = (1\text{-}x)^2 + 100(y - x^2)^2$



Comparison of Descent Methods

Method	Complexity	Convergence	Remarks
Steepest Descent	$O(n)$	Linear	Not used in practice; fails for many problems; slow convergence
Conjugate Gradient	$O(n)$	Linear	Reliable and computationally cheap but relatively slow convergence
Newton's	$O(n^3)$	Quadratic	Best convergence rate but requires Hessian information; not convergent for many problems; may converge to a maximum or saddle point; finding iteration step may be ill-conditioned
Damped Newton's	$O(n^3)$	Superlinear	Alleviates many of Newton's method drawbacks but still requires Hessian data
Quasi-Newton (DFP updating)	$O(n^2)$	Superlinear	May fail when used with soft line search
Quasi-Newton (BFGS updating)	$O(n^2)$	Superlinear	One of the best methods for unconstrained optimization to date

Bibliography

J. Nocedal, S.J. Wright, *Numerical Optimization*, Springer Science, 2006

A.R. Conn, N.I.M. Gould, and P.L. Toint, *Trust Region Methods*, MPS-SIAM Series on Optimization, 2000.

Exercise 1: Golden Ratio Search

Write a Matlab function that implements a golden ratio search algorithm. Let f be a function of a single parameter x. Let the (unique) minimum of f be bracketed by two points x_1 and x_2 ($x_1 < x_2$). Let $t = (1+5^{1/2})/2$ (golden ratio). Find two points x_3 and x_4 so that $(x_2 - x_3) = (x_2 - x_1)/t$ and $(x_4 - x_1) = (x_2 - x_1)/t$. Eliminate the point that has the largest value of $f(x_j)$, $j = 1, 2, 3, 4$. Set $x_1 = x_3$ (if x_1 is eliminated) or $x_2 = x_4$ (if x_2 is eliminated). Iterate the procedure until the minimum is located within the prescribed toletance.

Test the function using the following two cases:
1) $f(x) = (x - 2)^2$, $x_1 = 0$, $x_2 = 10$;
2) $f(x) = x^2 + 3\exp(-2x)$, $x_1 = 0$, $x_2 = 10$.
In both cases the tolerance should be set to 10^{-3}. Display the function value after each iteration of the algorithm.

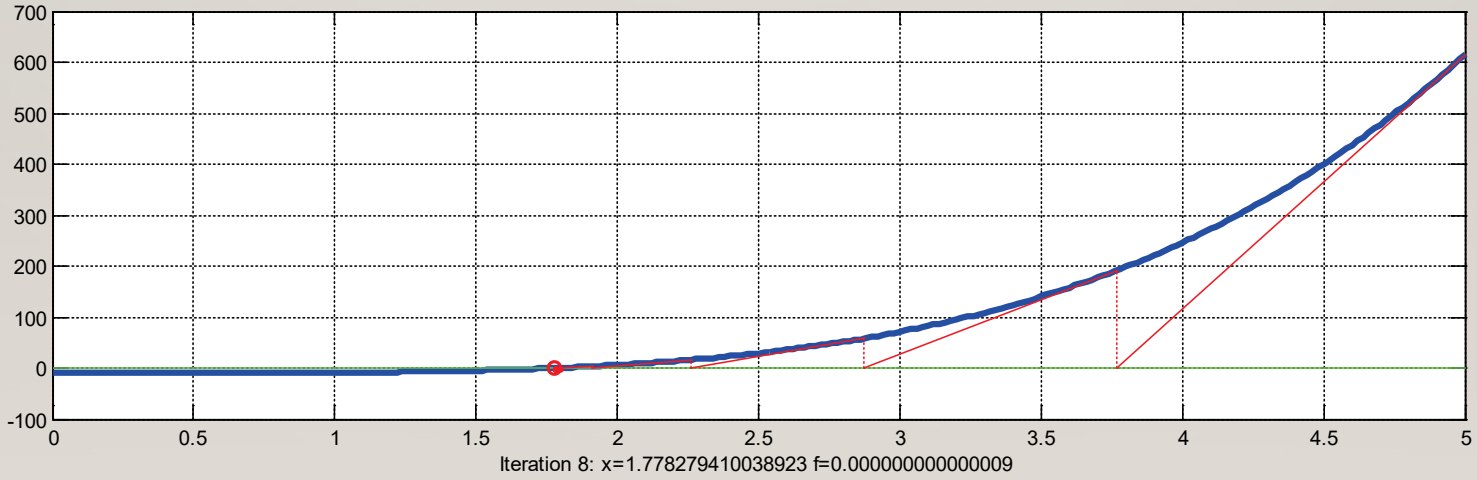
Exercise 2: Newton's Method

The root of the real-valued differentiable function f can be found using so-called Newton's method. Let x_0 be a reasonably good estimate of the root. The sequence $x_{n+1} = x_n - f(x_n)/f'(x_n)$, $n = 0, 1, ...,$ will then be convergent to x^* such that $f(x^*) = 0$.

Implement the Newton's method, visualize its operation, and test it for:

1) $f(x) = x^2$, $x_0 = 5$;
2) $f(x) = \ln(x)$, $x_0 = 0.1$;
3) $f(x) = x^4$, $x_0 = 5$;
4) $f(x) = x^{1/2} - 2$, $x_0 = 10$.

Visualization of the Newton's method for the third test case:



Exercise 3: Descent Method with Trust Region

Implement a descent method with trust region that uses a first-order model of the objective function. The model should be obtained by estimating the objective function gradient using finite differences. The model can be optimized using any available method (e.g., golden ratio search).

Test your function using:

1. $f(x) = x_1^2 + \ldots + x_n^2$ for different n (use $[1\ 2\ \ldots\ n]^T$ as a starting point),
2. $g(x) = \sin(x)$ (use 2 as a starting point),
3. $h(x) = \exp(x_1/5+x_2/2)+x_1^2+x_2^2$ (use $[5\ 3]^T$ as a starting point).
4. Rosenbrock function (use $[0\ 0]^T$ as a starting point).